Maitri Kasodariya
202001197
IT314
Lab 7

Section A:

Set of Equivalence classes:

Eq.C. 1 - date >=1 and date <= 31
Eq.C. 2 - date < 1
Eq.C. 3 - date > 31
Eq.C. 4 - month >= 1 and month <= 12
Eq.C. 5 - month < 1
Eq.C. 6 - month > 12
Eq.C. 7 - year >= 1900 and year <= 2015
Eq.C. 8 - year < 1900
Eq.C. 9 - year > 2015

The dates should not be invalid. Examples of invalid dates: 30/2/2012 , 60/3/2006, 00/0/0000

Equivalent test cases:

Sr.No.	Input	Expected Output
1	3/4/2012	3/3/2012
2	0/5/2002	Invalid
3	41/3/2011	Invalid
4	1/1/1999	31/12/1998
5	29/-4/2005	Invalid
6	20/16/2002	Invalid
7	15/6/1002	Invalid
8	5/6/2022	Invalid

Valid Dates:

Input: 2/2/2005
Expected output: 1/2/2005

Input: 1/1/2007
Expected Output: 31/12/2006

Invalid Dates:

31/4//2007
1/16/1999

Out of Range Date:

1/1/2018
1/1/1880

Boundary Values Analysis:

1. Earliest date - 1-1-1900
2. Last possible date - 31-12-2015
3. leap year - 29-2-2000
4. invalid leap year date - 29-2-2001
5. previous of earliest date - 31-12-1899
6. a day after latest date - 1-1-2016

Programs:

P1:The function linearSearch searches for a value v in an array of integers a. If v appears in the array a, then the function returns the first index i, such that a[i] == v; otherwise, -1 is returned.

```
int linearSearch(int v, int a[])
{
      int i = 0;
      while (i < a.length)
      {
            if (a[i] == v)
                  return(i);
            i++;
      }
      return (-1);
}
```

Equivalence Partitioning

if v is present in a then return index i where a[i]==v
if v is not present then return -1

Boundary Value Analysis

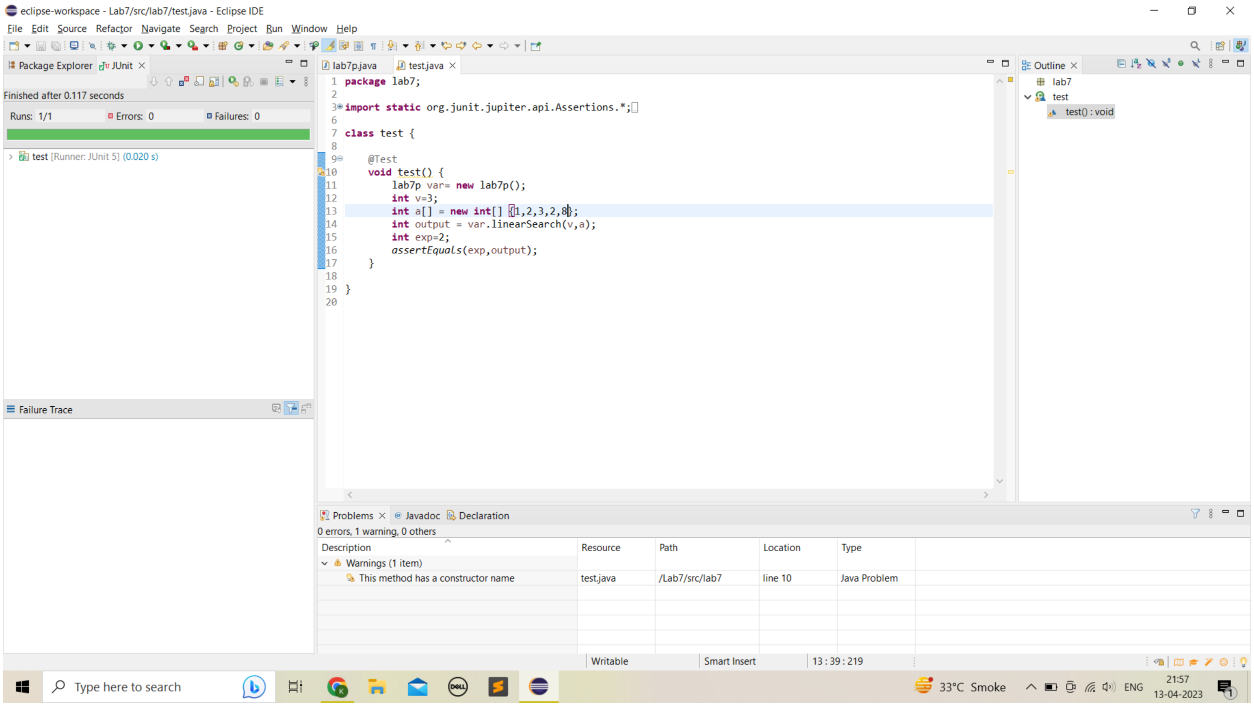
If present at i=0 then return 0

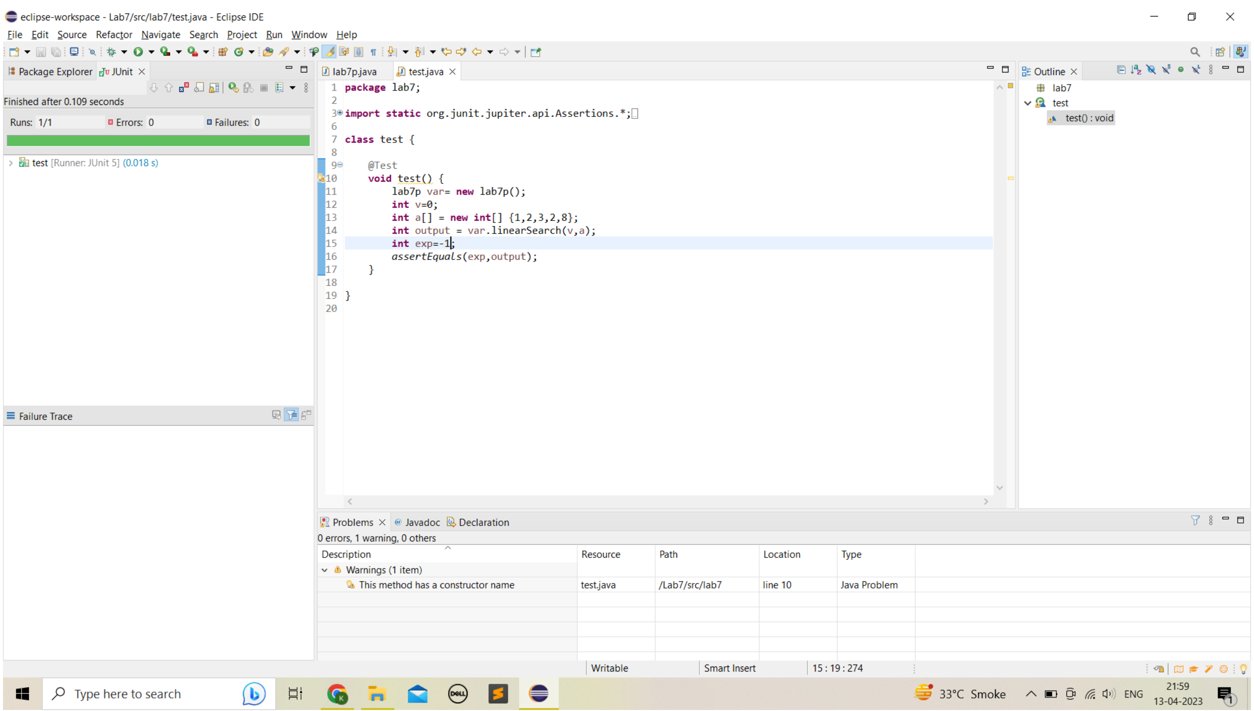
If empty array then return -1

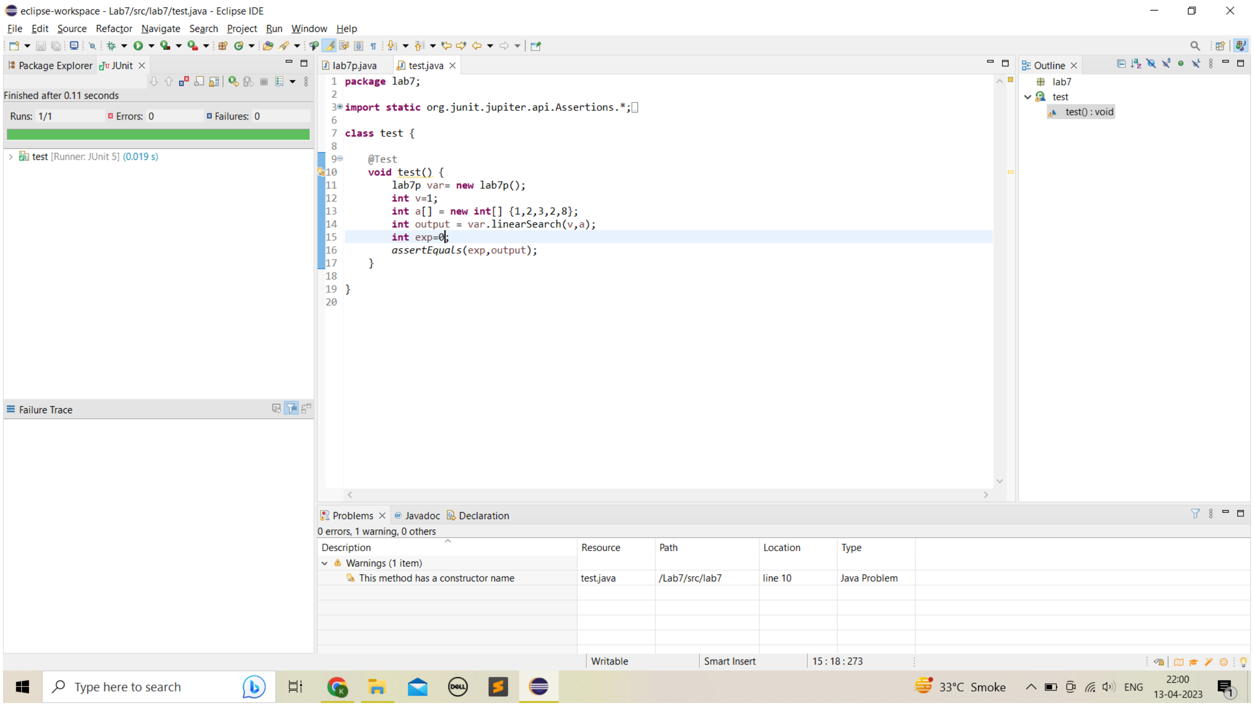
If v not present then return -1

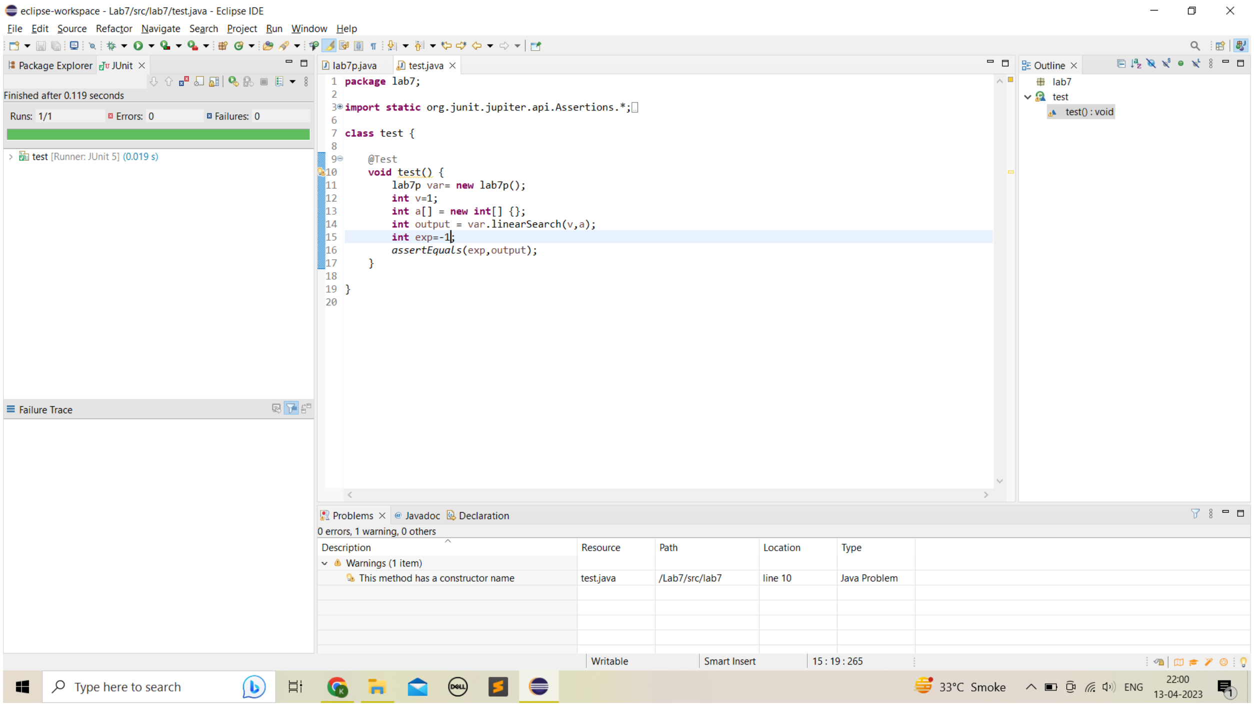
Testcase:

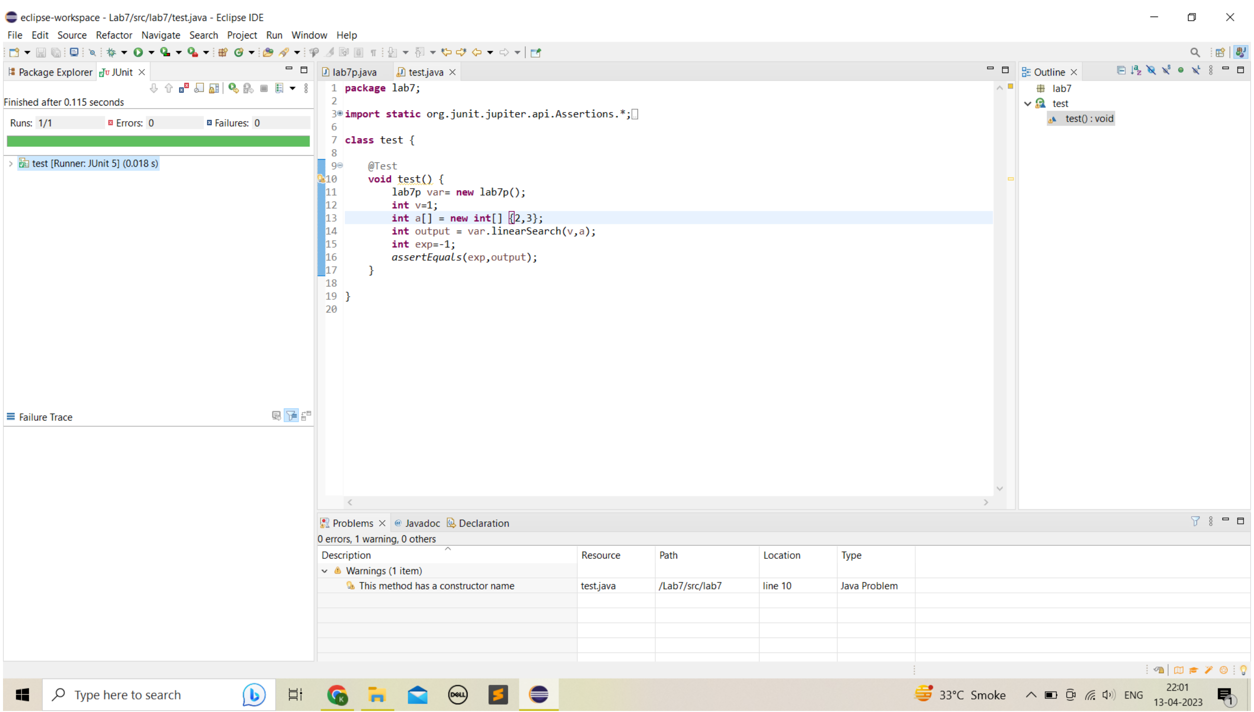
Input	Expected Output
v=3, a={1,2,3,2,8}	2
v=0,a={1,2,3,2,8}	-1
v=1,a={1,2,3,2,8}	0
v=1,a={}	-1
v=1,a={2,3}	-1


eclipse-workspace - Lab7/src/lab7/test.java - Eclipse IDE
File Edit Source Refactor Navigate Search Project Run Window Help
Package Explorer JUnit
Finished after 0.117 seconds
Runs: 1/1 Errors: 0 Failures: 0
test [Runner: JUnit 5] (0.020 s)
Failure Trace
lab7p.java test.java
package lab7;
import static org.junit.jupiter.api.Assertions.*;
class test {
@Test
void test() {
lab7p var= new lab7p();
int v=3;
int a[] = new int[] {1,2,3,2,8};
int output = var.linearSearch(v,a);
int exp=2;
assertEquals(exp,output);
}
}
Outline
lab7
test
test() : void
Problems Javadoc Declaration
0 errors, 1 warning, 0 others
Description Resource Path Location Type
Warnings (1 item)
This method has a constructor name test.java /Lab7/src/lab7 line 10 Java Problem
Writable Smart Insert 13 : 39 : 219
Type here to search
33°C Smoke ENG 21:57 13-04-2023


eclipse-workspace - Lab7/src/lab7/test.java - Eclipse IDE
File Edit Source Refactor Navigate Search Project Run Window Help
Package Explorer JUnit
Finished after 0.109 seconds
Runs: 1/1 Errors: 0 Failures: 0
test [Runner: JUnit 5] (0.018 s)
Failure Trace
lab7p.java test.java
package lab7;
import static org.junit.jupiter.api.Assertions.*;
class test {
@Test
void test() {
lab7p var= new lab7p();
int v=0;
int a[] = new int[] {1,2,3,2,8};
int output = var.linearSearch(v,a);
int exp=-1;
assertEquals(exp,output);
}
}
Outline
lab7
test
test() : void
Problems Javadoc Declaration
0 errors, 1 warning, 0 others
Description Resource Path Location Type
Warnings (1 item)
This method has a constructor name test.java /Lab7/src/lab7 line 10 Java Problem
Writable Smart Insert 15 : 19 : 274
Type here to search
33°C Smoke ENG 21:59 13-04-2023


eclipse-workspace - Lab7/src/lab7/test.java - Eclipse IDE
File Edit Source Refactor Navigate Search Project Run Window Help
Package Explorer
JUnit
Finished after 0.11 seconds
Runs: 1/1
Errors: 0
Failures: 0
test [Runner: JUnit 5] (0.019 s)
Failure Trace
lab7p.java
test.java
package lab7;
import static org.junit.jupiter.api.Assertions.*;
class test {
@Test
void test() {
lab7p var= new lab7p();
int v=1;
int a[] = new int[] {1,2,3,2,8};
int output = var.linearSearch(v,a);
int exp=0;
assertEquals(exp,output);
}
}
Outline
lab7
test
test() : void
Problems
Javadoc
Declaration
0 errors, 1 warning, 0 others
Description
Resource
Path
Location
Type
Warnings (1 item)
This method has a constructor name
test.java
/Lab7/src/lab7
line 10
Java Problem
Writable
Smart Insert
15 : 18 : 273
Type here to search
33°C Smoke
ENG
22:00
13-04-2023





P2:The function countItem returns the number of times a value v appears in an array of integers a.

```
int countItem(int v, int a[])
```

[redacted]

```
{
        int count = 0;
        for (int i = 0; i < a.length; i++)
        {
              if (a[i] == v)
                    count++;
        }
        return (count);
}
```

Equivalence Partitioning

if v is present in a then return number of times v appears
if v is not present then return 0

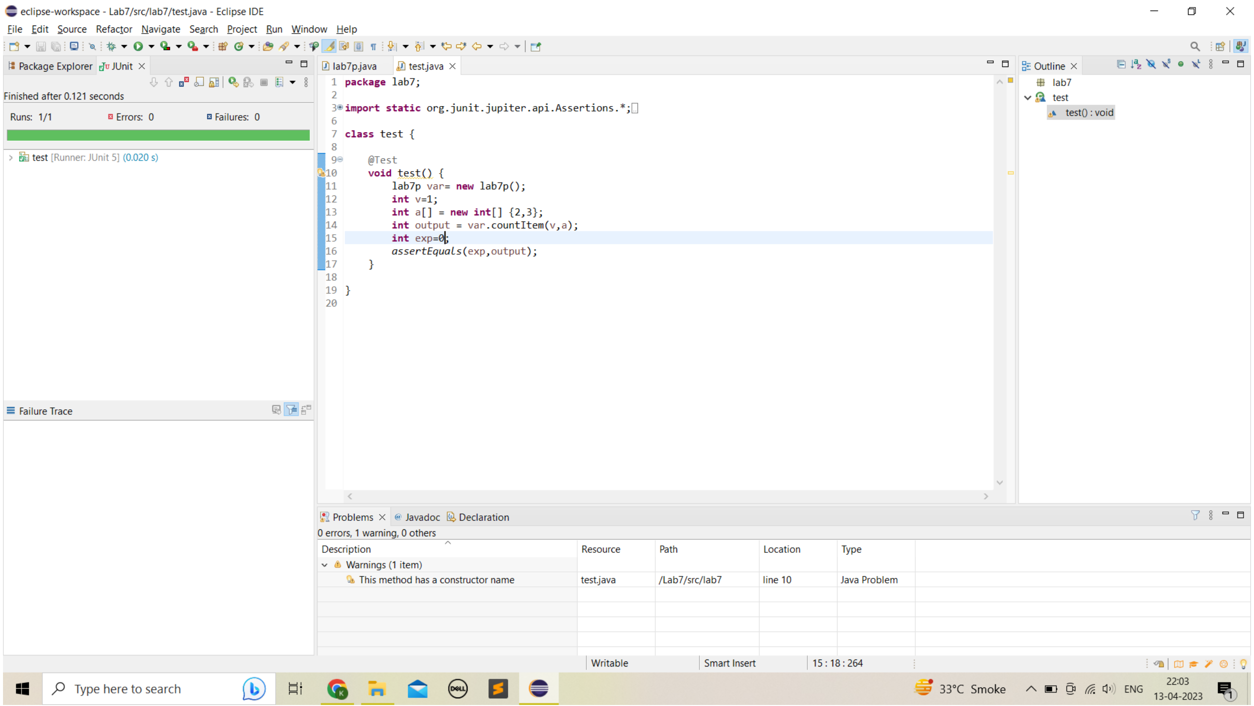
Boundary Value Analysis

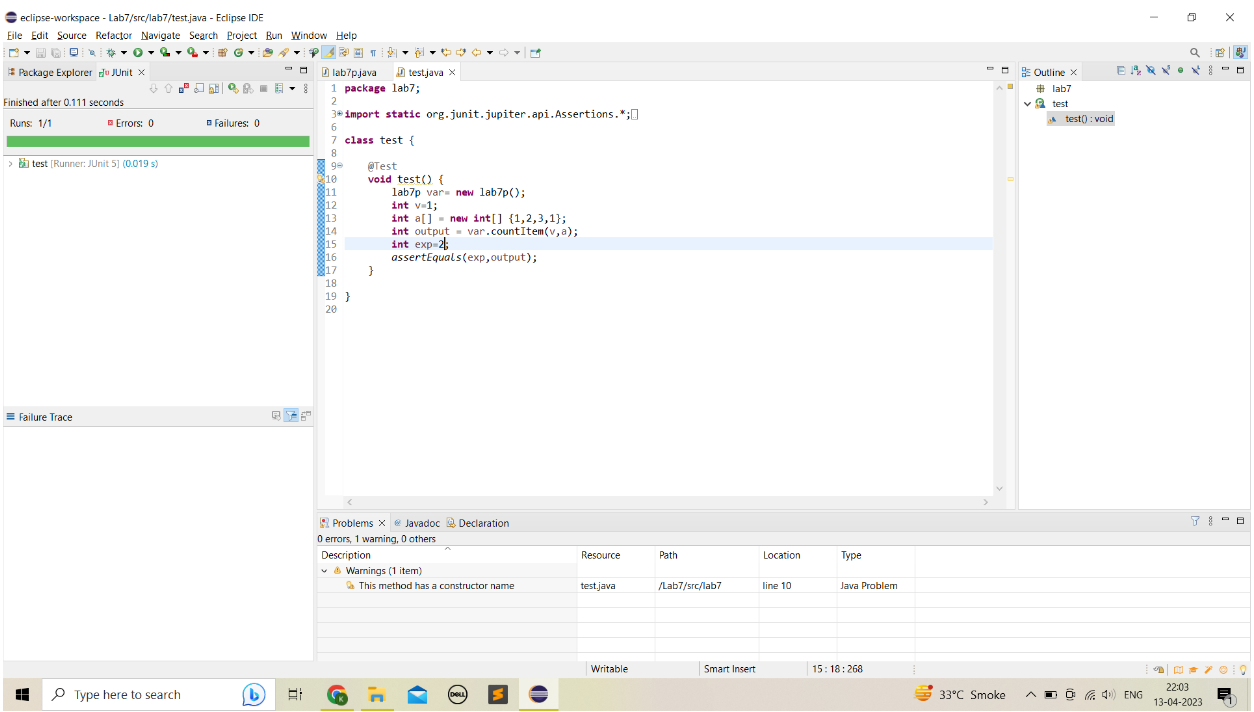
If empty array then return 0
If v not present then return 0
If present at i=0 then return 1
If present multiple times then return count

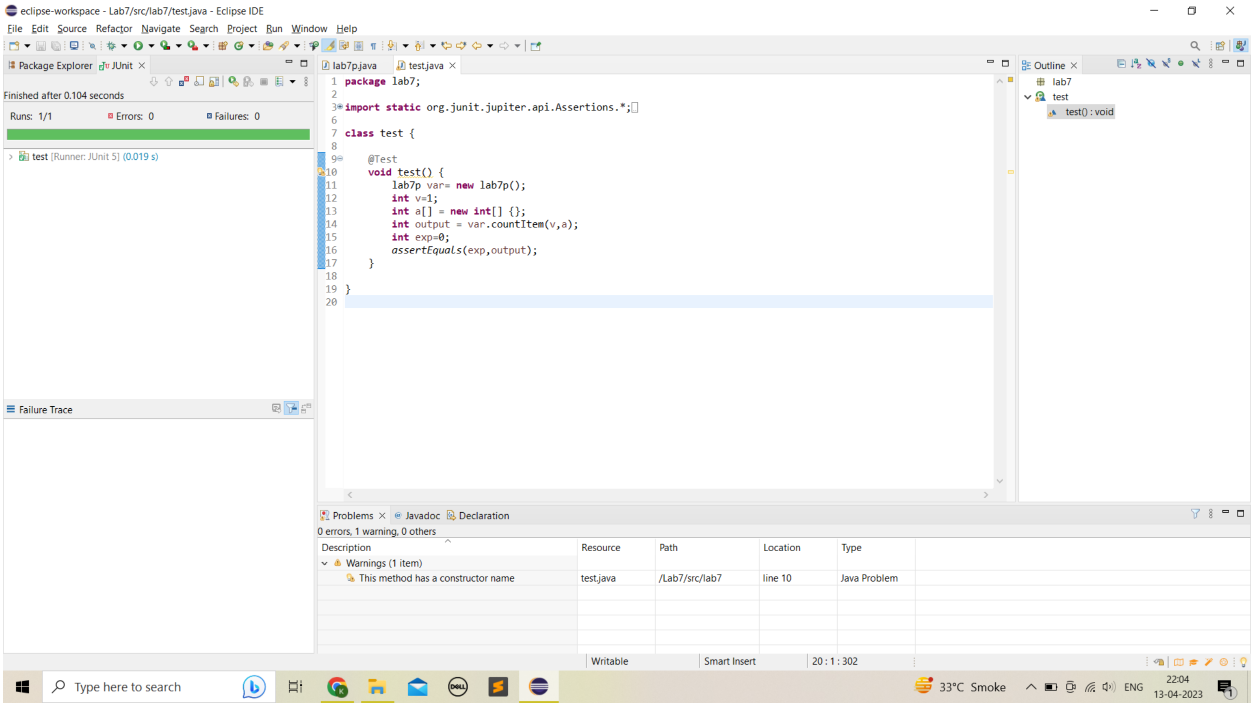
Testcase:

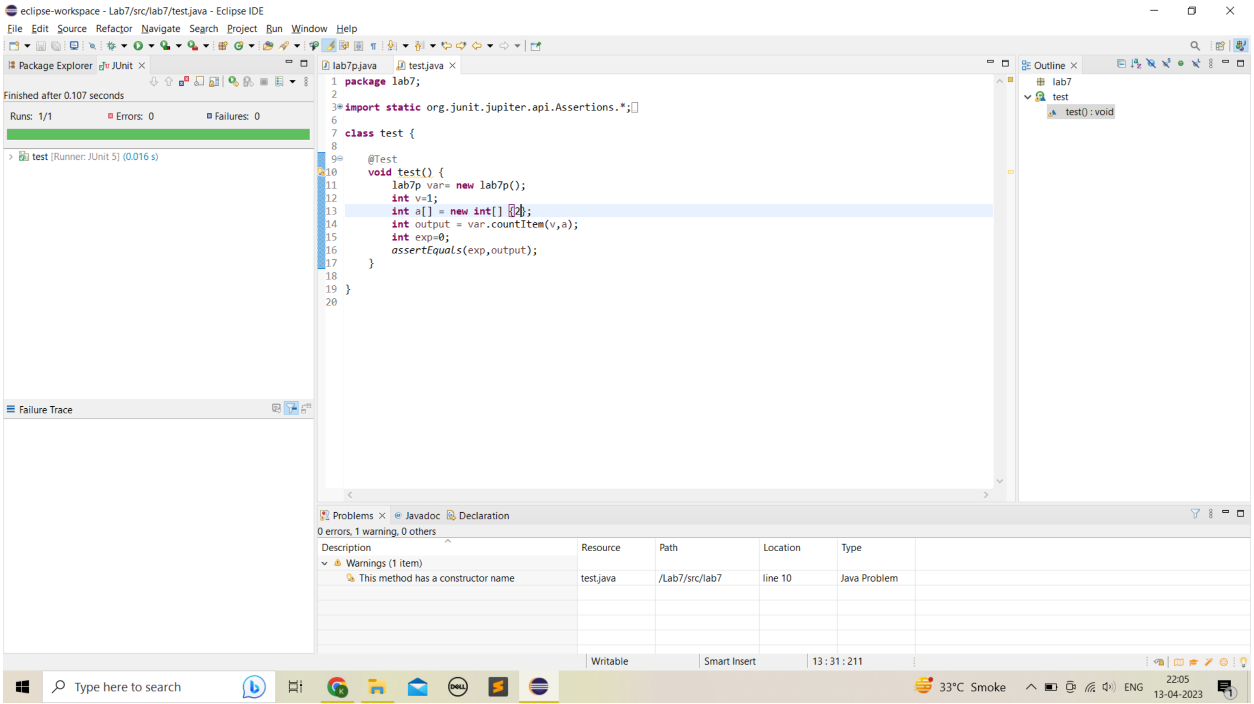
Input	Expected Output
v=1, a={2,3}	0
v=1,a={1,2,3,1}	2
v=1,a={}	0

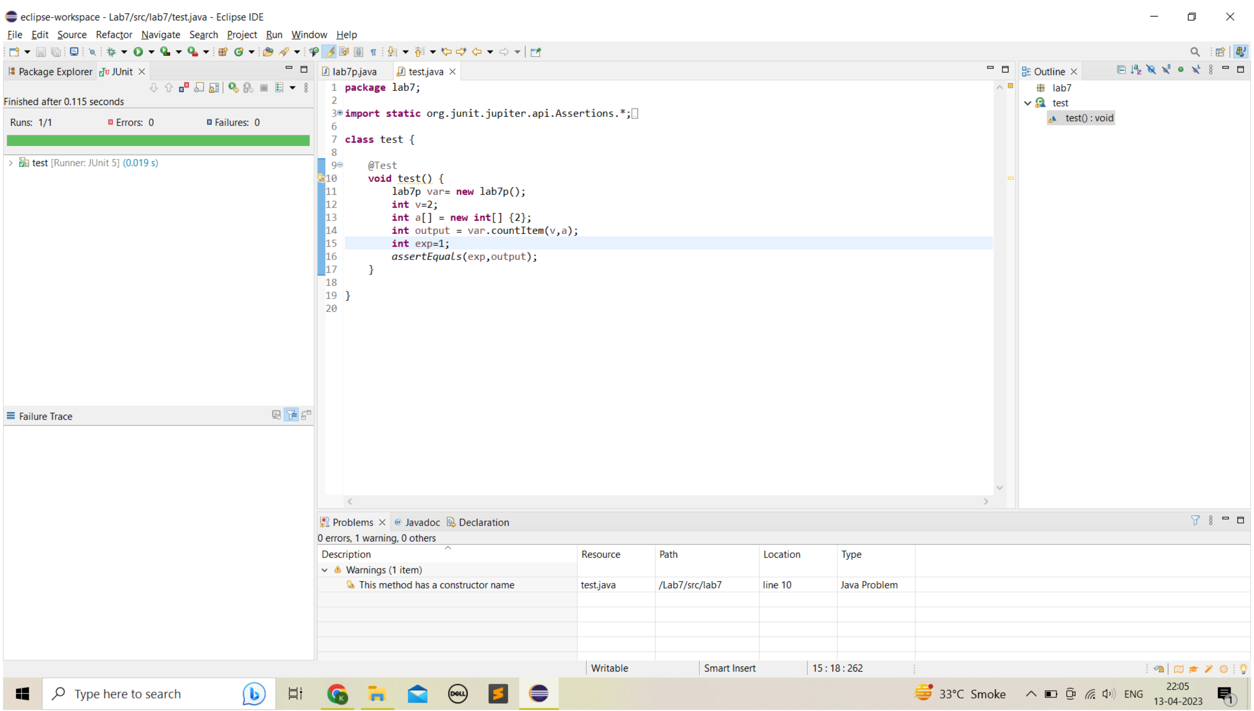
v=1,a={2}	0
v=2,a={2}	1


eclipse-workspace - Lab7/src/lab7/test.java - Eclipse IDE
File Edit Source Refactor Navigate Search Project Run Window Help
Package Explorer JUnit
Finished after 0.121 seconds
Runs: 1/1 Errors: 0 Failures: 0
test [Runner: JUnit 5] (0.020 s)
Failure Trace
lab7p.java test.java
package lab7;
import static org.junit.jupiter.api.Assertions.*;
class test {
@Test
void test() {
lab7p var= new lab7p();
int v=1;
int a[] = new int[] {2,3};
int output = var.countItem(v,a);
int exp=0;
assertEquals(exp,output);
}
}
Outline
lab7
test
test() : void
Problems Javadoc Declaration
0 errors, 1 warning, 0 others
Description Resource Path Location Type
Warnings (1 item)
This method has a constructor name test.java /Lab7/src/lab7 line 10 Java Problem
Writable Smart Insert 15 : 18 : 264
Type here to search
33°C Smoke ENG 22:03 13-04-2023


eclipse-workspace - Lab7/src/lab7/test.java - Eclipse IDE
File Edit Source Refactor Navigate Search Project Run Window Help
Package Explorer JUnit
Finished after 0.111 seconds
Runs: 1/1 Errors: 0 Failures: 0
test [Runner: JUnit 5] (0.019 s)
Failure Trace
lab7p.java test.java
package lab7;
import static org.junit.jupiter.api.Assertions.*;
class test {
@Test
void test() {
lab7p var= new lab7p();
int v=1;
int a[] = new int[] {1,2,3,1};
int output = var.countItem(v,a);
int exp=2;
assertEquals(exp,output);
}
}
Outline
lab7
test
test() : void
Problems Javadoc Declaration
0 errors, 1 warning, 0 others
Description Resource Path Location Type
Warnings (1 item)
This method has a constructor name test.java /Lab7/src/lab7 line 10 Java Problem
Writable Smart Insert 15 : 18 : 268
Type here to search
33°C Smoke ENG 22:03 13-04-2023


eclipse-workspace - Lab7/src/lab7/test.java - Eclipse IDE
File Edit Source Refactor Navigate Search Project Run Window Help
Package Explorer
JUnit
Finished after 0.104 seconds
Runs: 1/1
Errors: 0
Failures: 0
test [Runner: JUnit 5] (0.019 s)
Failure Trace
lab7p.java
test.java
package lab7;
import static org.junit.jupiter.api.Assertions.*;
class test {
@Test
void test() {
lab7p var= new lab7p();
int v=1;
int a[] = new int[] {};
int output = var.countItem(v,a);
int exp=0;
assertEquals(exp,output);
}
}
Outline
lab7
test
test() : void
Problems
Javadoc
Declaration
0 errors, 1 warning, 0 others
Description
Resource
Path
Location
Type
Warnings (1 item)
This method has a constructor name
test.java
/Lab7/src/lab7
line 10
Java Problem
Writable
Smart Insert
20 : 1 : 302
Type here to search
33°C Smoke
ENG
22:04
13-04-2023





P3:The function binarySearch searches for a value v in an ordered array of integers a. If v appears in
the array a, then the function returns an index i, such that a[i] == v; otherwise, -1 is returned.

Assumption: the elements in the array a are sorted in non-decreasing order.

```
int binarySearch(int v, int a[])
{
      int lo,mid,hi;
      lo = 0;
      hi = a.length-1;
      while (lo <= hi)
      {
            mid = (lo+hi)/2;
            if (v == a[mid])
                  return (mid);
            else if (v < a[mid])
                  hi = mid-1;
            else
                  lo = mid+1;
      }
      return(-1);
}
```

<u>Equivalence Partitioning</u>

if v is present in a then return index of v
if v is not present then return -1

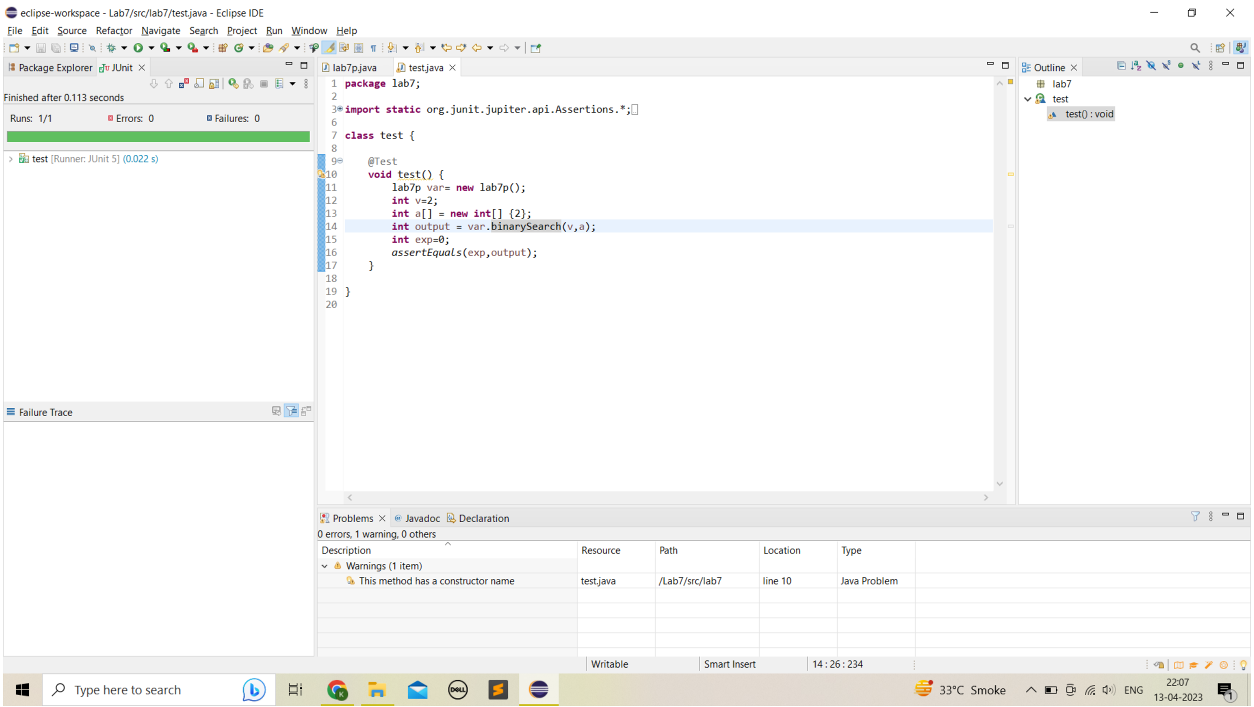
<u>Boundary Value Analysis</u>

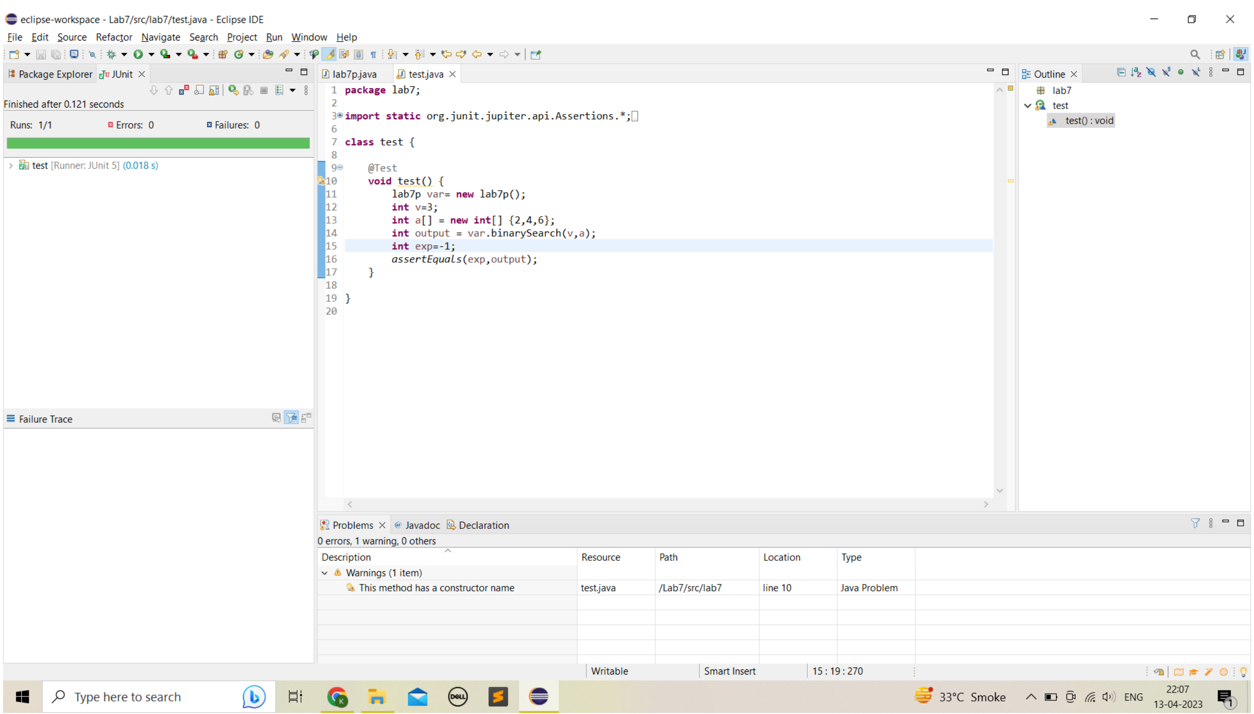
empty array then return -1
not present then return -1
present at i=0 then return 0

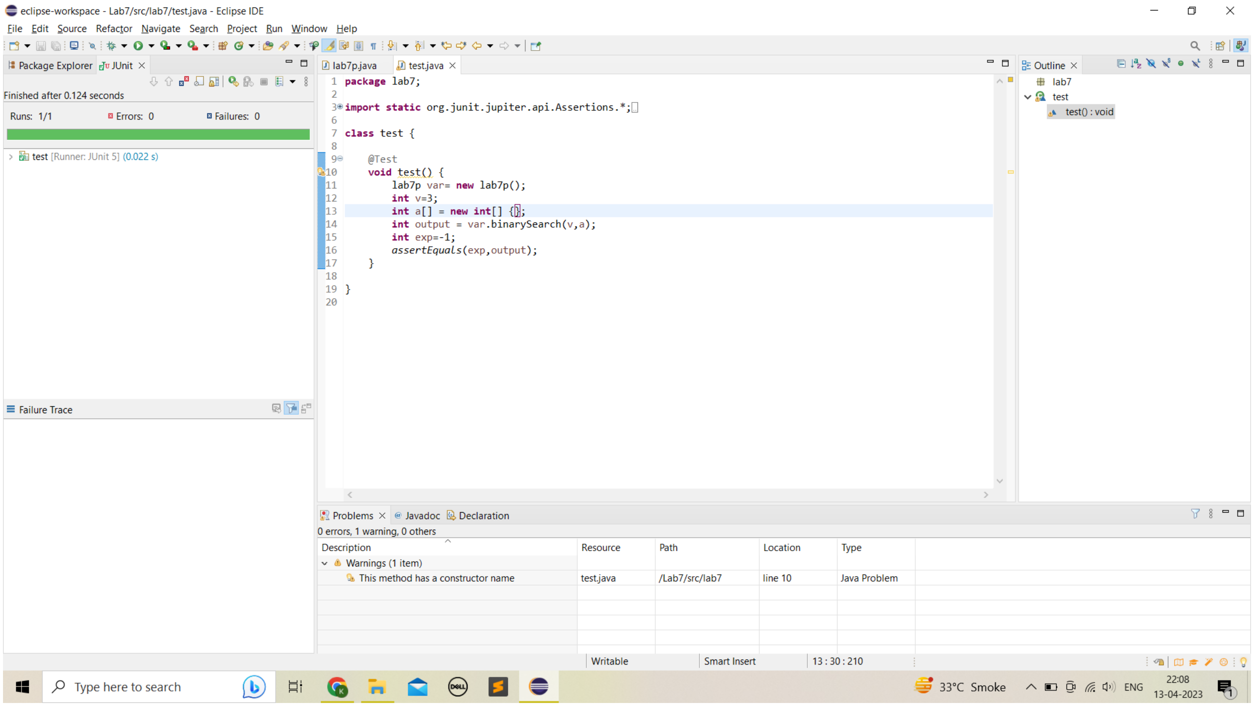
Testcase:

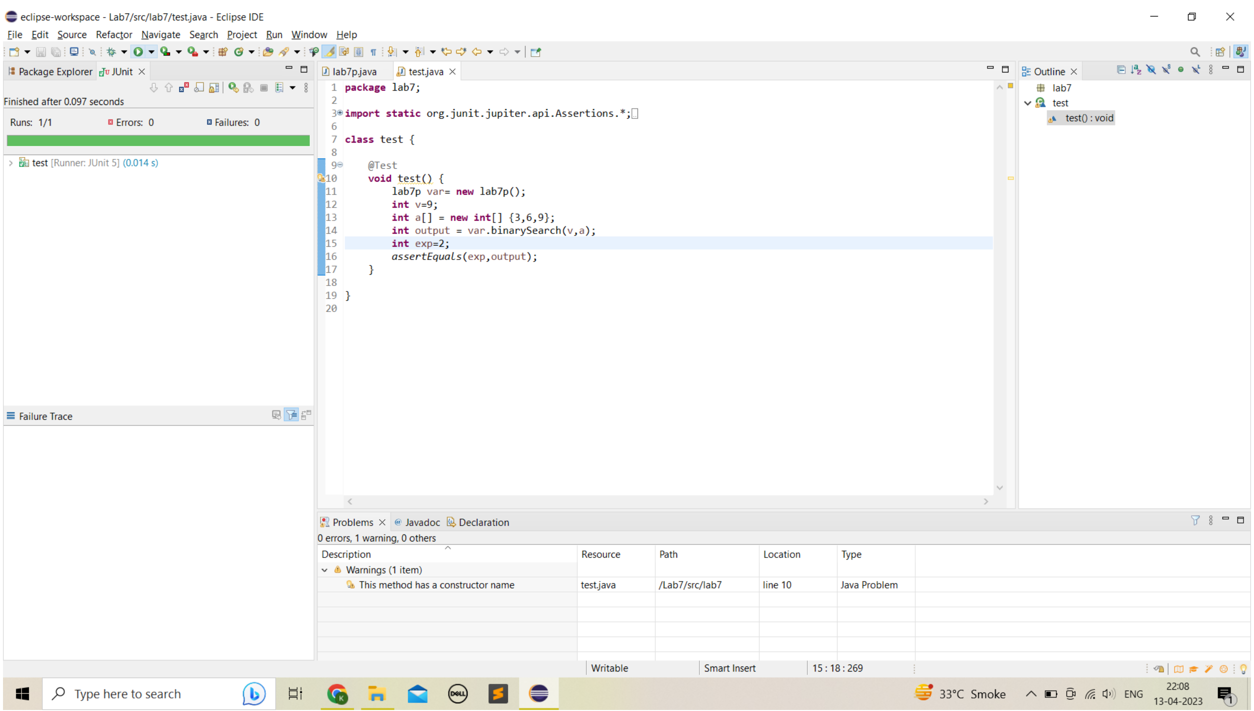
Input	Expected Output
v=2, a={2}	0

v=3,a={2,4,6}	-1
v=3,a={}	-1
v=9,a={3,6,9}	2









P4:The following problem has been adapted from The Art of Software Testing, by G. Myers (1979). The
function triangle takes three integer parameters that are interpreted as the lengths of the sides of a
triangle. It returns whether the triangle is equilateral (three lengths equal), isosceles (two lengths equal),

scalene (no lengths equal), or invalid (impossible lengths).

```
final int EQUILATERAL = 0;
final int ISOSCELES = 1;
final int SCALENE = 2;
final int INVALID = 3;
int triangle(int a, int b, int c)
{
      if (a >= b+c || b >= a+c || c >= a+b)
            return(INVALID);
      if (a == b && b == c)
            return(EQUILATERAL);
      if (a == b || a == c || b == c)
            return(ISOSCELES);
      return(SCALENE);
```

Equivalence Partitioning

a+b<=c then INVALID
a=b=c then Equilateral Triangle
a=b && a!= c then Isosceles Triangle
a!= b != c then Scalene Triangle

Boundary Value Analysis

a<=0,b<=0,c<=0 then Invalid

a+b<=c then Invalid

b+c<=a then Invalid

c+a<=b then Invalid

a=b=c then Equilateral

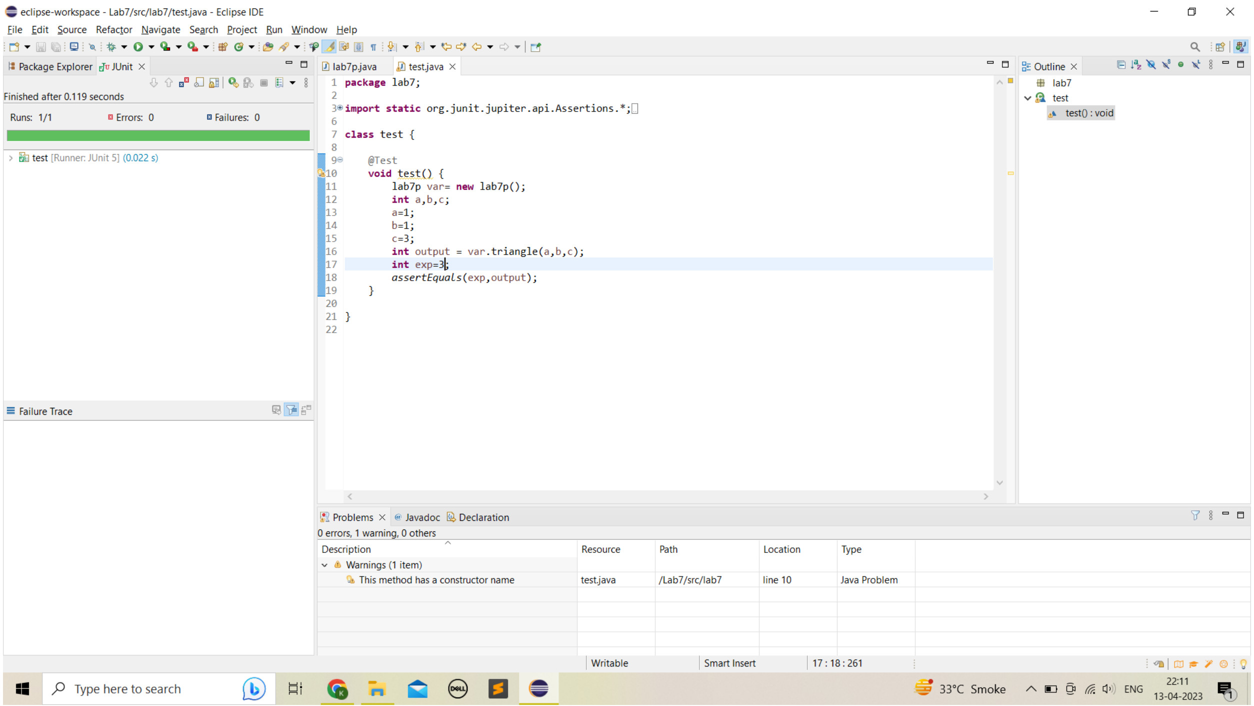
a=b , a!= c then Isosceles

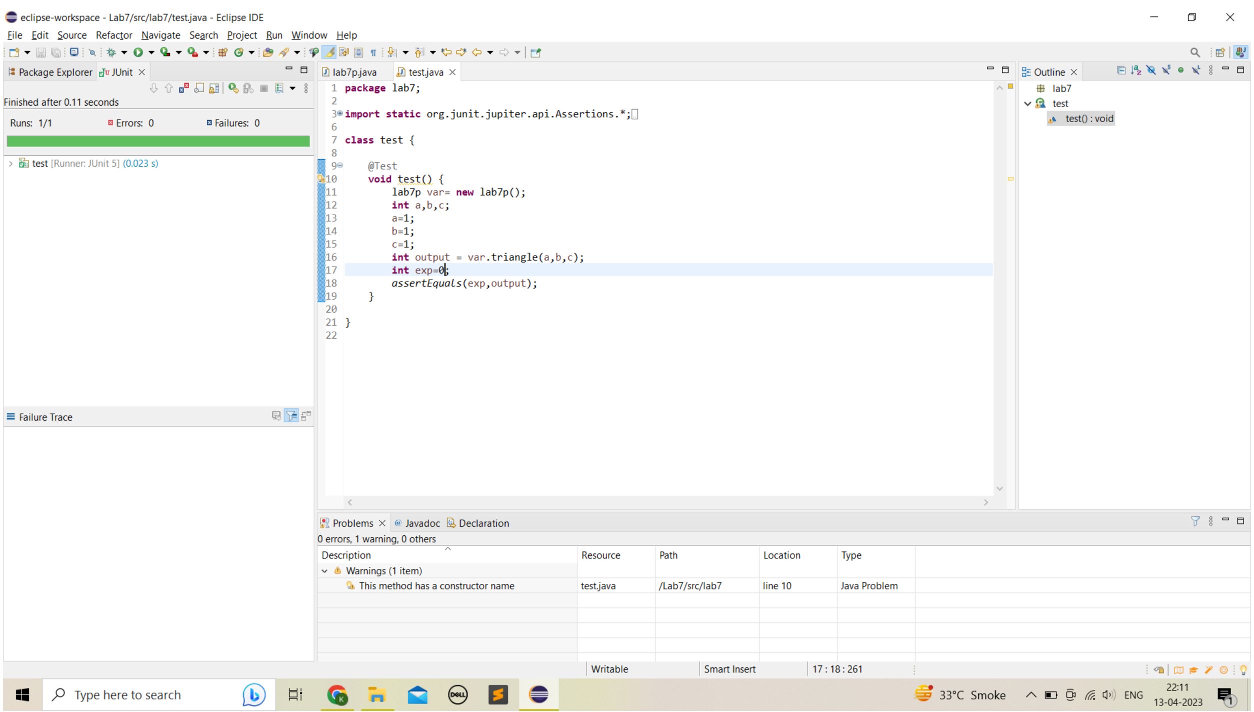
b=c , b!= a then Isosceles

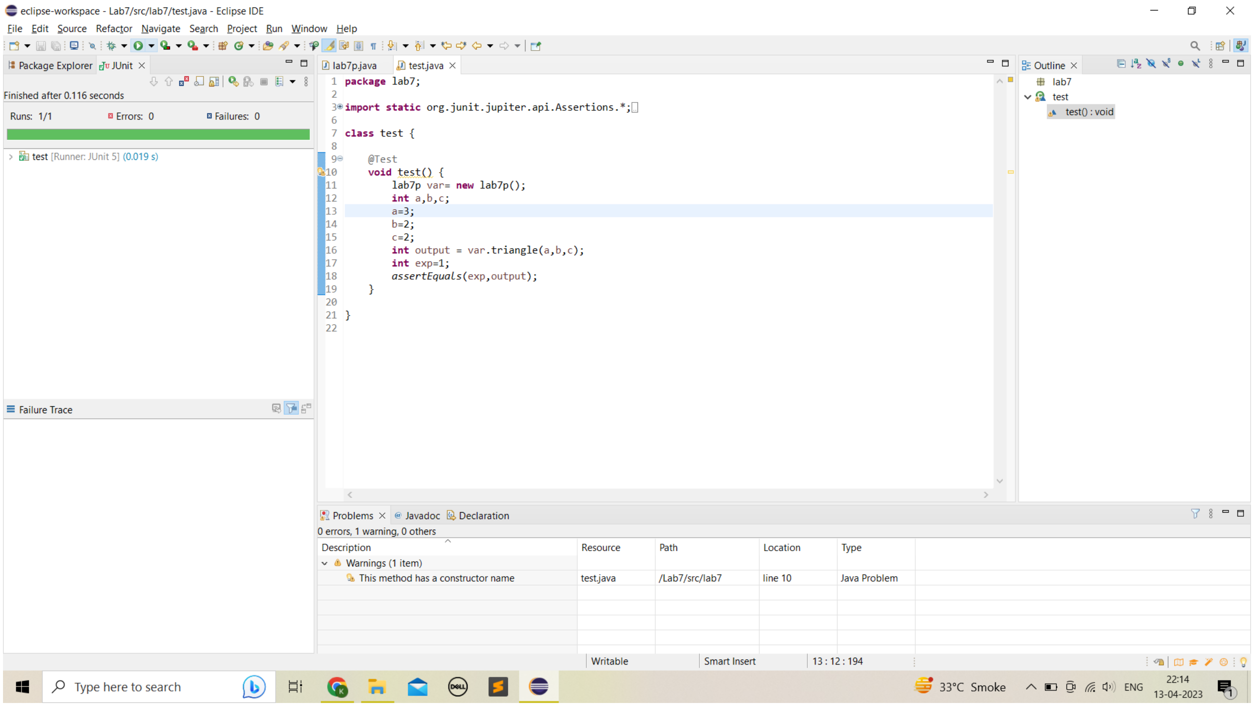
c=a , a!= b then Isosceles

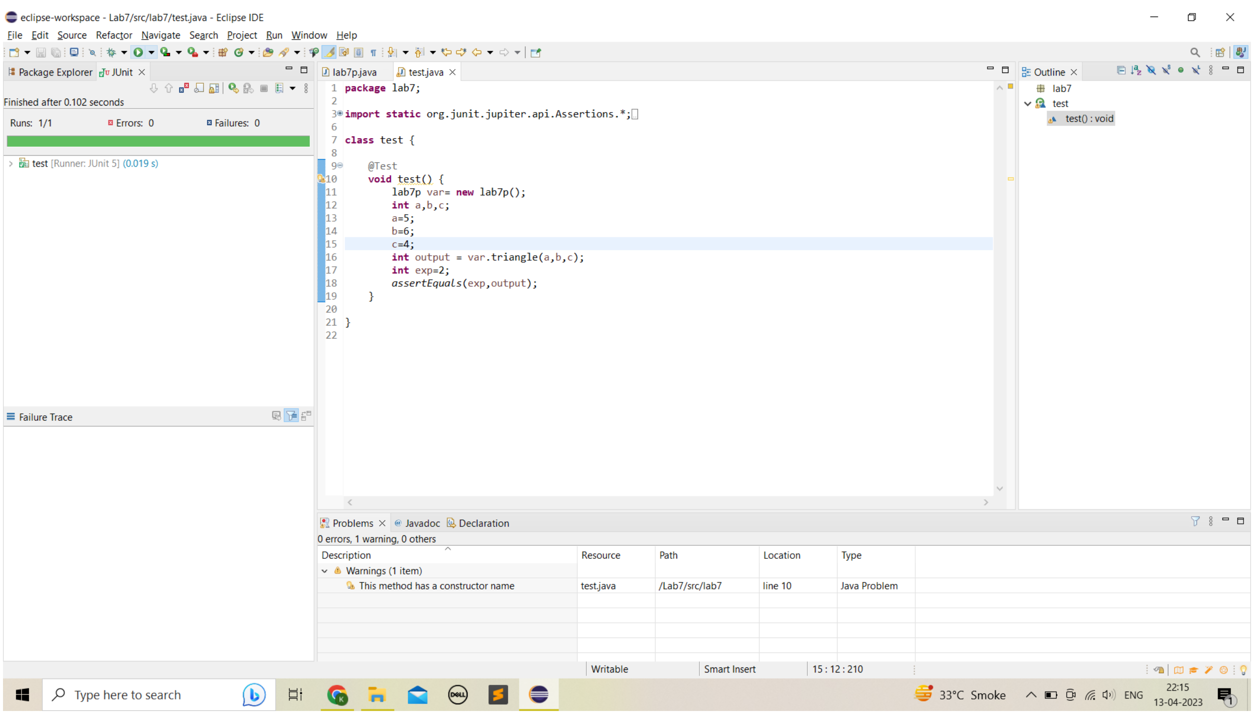
a!= b!=c then Scalene

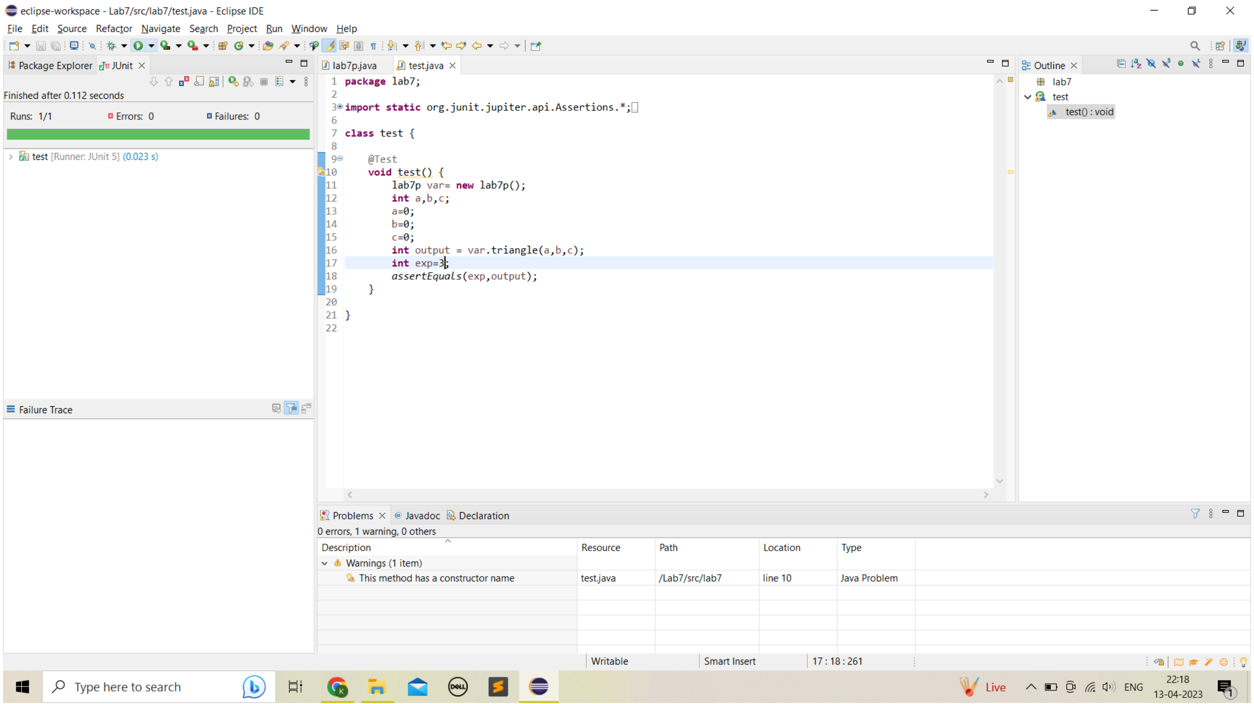
a	b	c	Expected output
1	1	3	3
1	1	1	0
3	2	2	1
5	6	4	2
0	0	0	3
1	-1	1	3

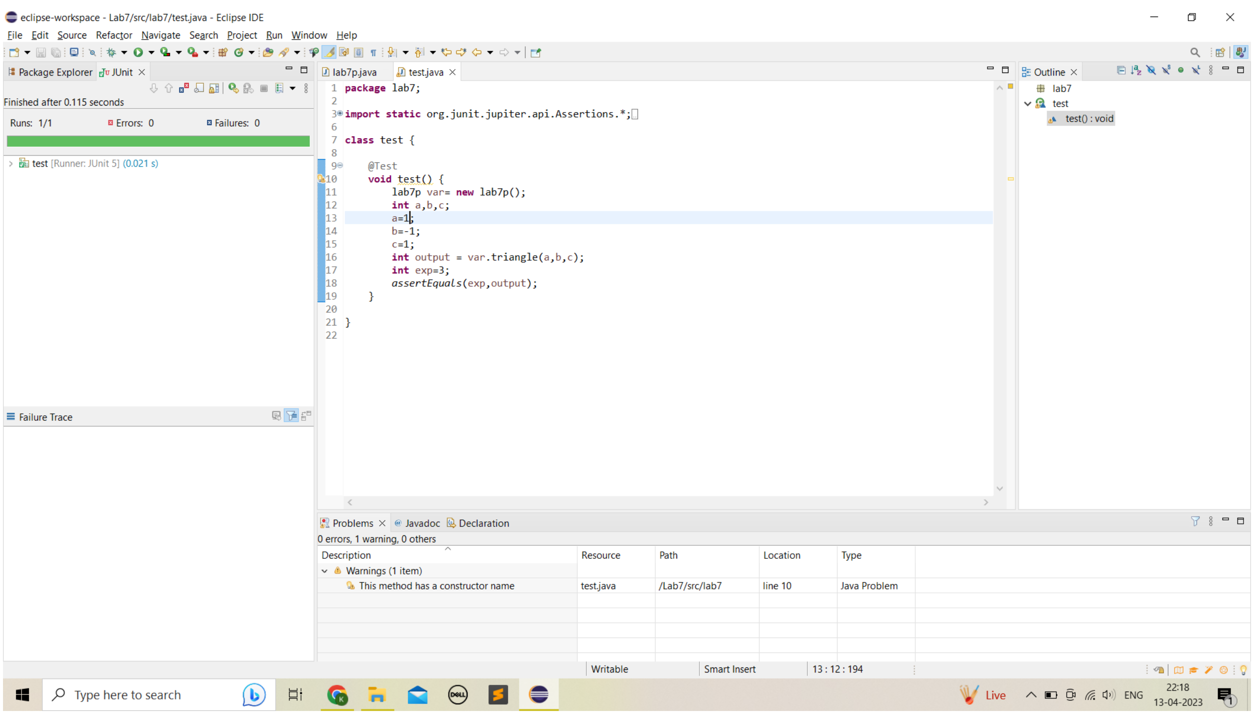

eclipse-workspace - Lab7/src/lab7/test.java - Eclipse IDE
File Edit Source Refactor Navigate Search Project Run Window Help
Package Explorer JUnit
Finished after 0.119 seconds
Runs: 1/1 Errors: 0 Failures: 0
test [Runner: JUnit 5] (0.022 s)
Failure Trace
lab7p.java test.java
package lab7;
import static org.junit.jupiter.api.Assertions.*;
class test {
@Test
void test() {
lab7p var= new lab7p();
int a,b,c;
a=1;
b=1;
c=3;
int output = var.triangle(a,b,c);
int exp=3;
assertEquals(exp,output);
}
}
Outline
lab7
test
test() : void
Problems Javadoc Declaration
0 errors, 1 warning, 0 others
Description Resource Path Location Type
Warnings (1 item)
This method has a constructor name test.java /Lab7/src/lab7 line 10 Java Problem
Writable Smart Insert 17 : 18 : 261
Type here to search
33°C Smoke ENG 22:11 13-04-2023


eclipse-workspace - Lab7/src/lab7/test.java - Eclipse IDE
File Edit Source Refactor Navigate Search Project Run Window Help
Package Explorer JUnit
Finished after 0.11 seconds
Runs: 1/1 Errors: 0 Failures: 0
test [Runner: JUnit 5] (0.023 s)
Failure Trace
lab7p.java test.java
package lab7;
import static org.junit.jupiter.api.Assertions.*;
class test {
@Test
void test() {
lab7p var= new lab7p();
int a,b,c;
a=1;
b=1;
c=1;
int output = var.triangle(a,b,c);
int exp=0;
assertEquals(exp,output);
}
}
Outline
lab7
test
test() : void
Problems Javadoc Declaration
0 errors, 1 warning, 0 others
Description Resource Path Location Type
Warnings (1 item)
This method has a constructor name test.java /Lab7/src/lab7 line 10 Java Problem
Writable Smart Insert 17 : 18 : 261
Type here to search
33°C Smoke ENG 22:11 13-04-2023


eclipse-workspace - Lab7/src/lab7/test.java - Eclipse IDE
File Edit Source Refactor Navigate Search Project Run Window Help
Package Explorer
JUnit
Finished after 0.116 seconds
Runs: 1/1
Errors: 0
Failures: 0
test [Runner: JUnit 5] (0.019 s)
Failure Trace
lab7p.java
test.java
package lab7;
import static org.junit.jupiter.api.Assertions.*;
class test {
@Test
void test() {
lab7p var= new lab7p();
int a,b,c;
a=3;
b=2;
c=2;
int output = var.triangle(a,b,c);
int exp=1;
assertEquals(exp,output);
}
}
Outline
lab7
test
test() : void
Problems
Javadoc
Declaration
0 errors, 1 warning, 0 others
Description
Resource
Path
Location
Type
Warnings (1 item)
This method has a constructor name
test.java
/Lab7/src/lab7
line 10
Java Problem
Writable
Smart Insert
13 : 12 : 194
Type here to search
33°C Smoke
ENG
22:14
13-04-2023


eclipse-workspace - Lab7/src/lab7/test.java - Eclipse IDE
File Edit Source Refactor Navigate Search Project Run Window Help
Package Explorer
JUnit
Finished after 0.102 seconds
Runs: 1/1
Errors: 0
Failures: 0
test [Runner: JUnit 5] (0.019 s)
Failure Trace
lab7p.java
test.java
package lab7;
import static org.junit.jupiter.api.Assertions.*;
class test {
@Test
void test() {
lab7p var= new lab7p();
int a,b,c;
a=5;
b=6;
c=4;
int output = var.triangle(a,b,c);
int exp=2;
assertEquals(exp,output);
}
}
Outline
lab7
test
test() : void
Problems
Javadoc
Declaration
0 errors, 1 warning, 0 others
Description
Resource
Path
Location
Type
Warnings (1 item)
This method has a constructor name
test.java
/Lab7/src/lab7
line 10
Java Problem
Writable
Smart Insert
15 : 12 : 210
Type here to search
33°C Smoke
ENG
22:15
13-04-2023





P5:The function prefix (String s1, String s2) returns whether or not the string s1 is a prefix of string s2 (you may assume that neither s1 nor s2 is null).

```
public static boolean prefix(String s1, String s2)
{
	if (s1.length() > s2.length())
	{
		return false;
	}
	for (int i = 0; i < s1.length(); i++)
	{
		if (s1.charAt(i) != s2.charAt(i))
		{
			return false;
		}
	}
	return true;
}
```

Equivalence Partitioning

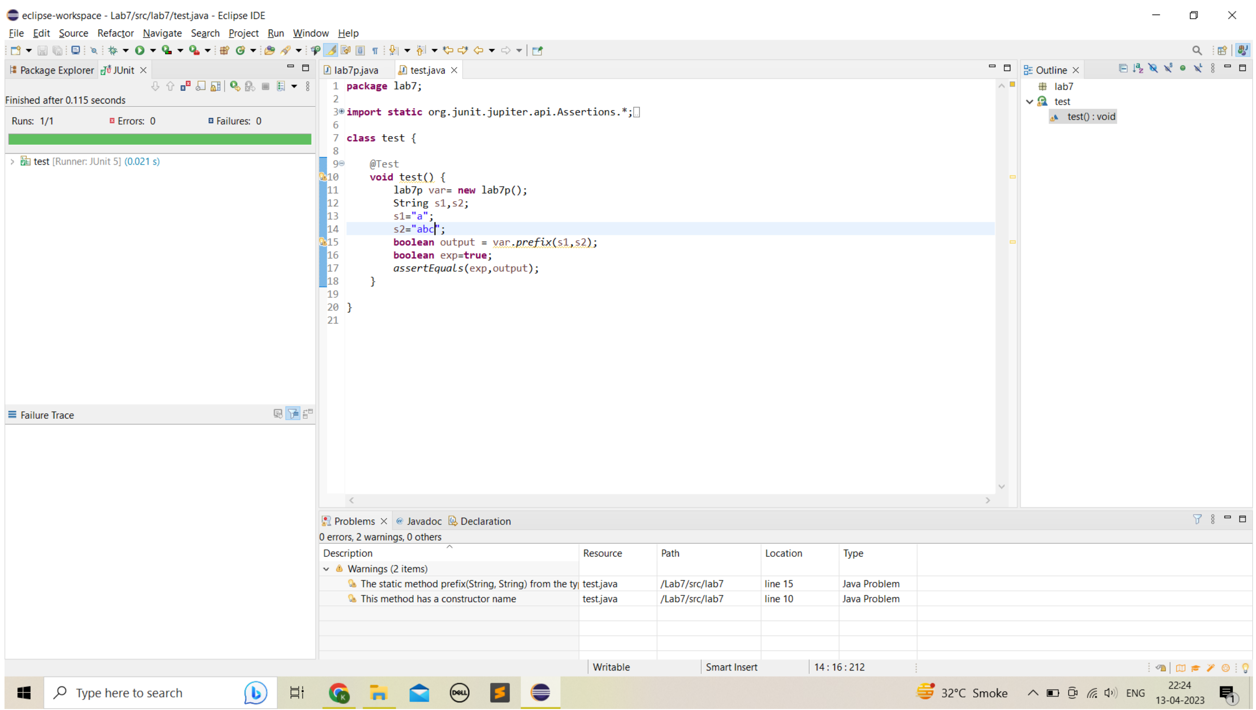
empty s1 and s2 then true
empty s1 then true
s1 prefix of s2 then true
s1 not prefix of s2 then false
s1 longer than s2 then false

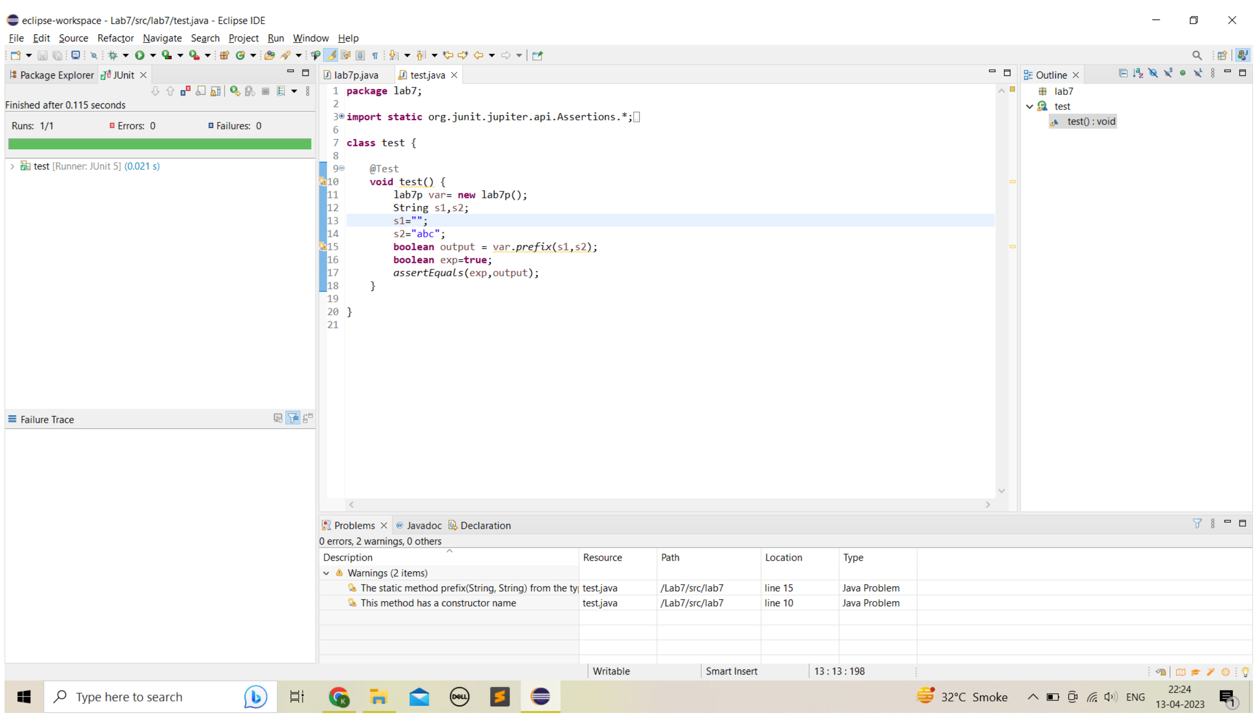
Boundary value analysis

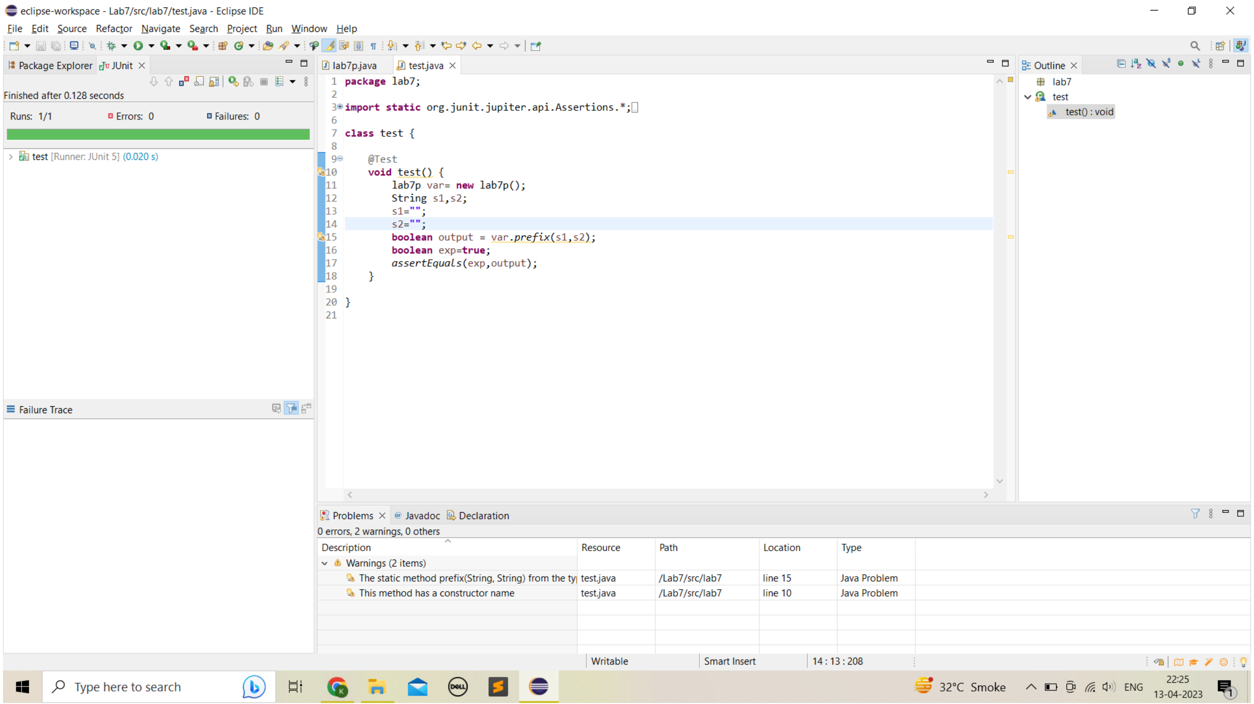
empty s1 and s2 then true
empty s1 then true
s1 prefix of s2 then true
s1 longer than s2 then false

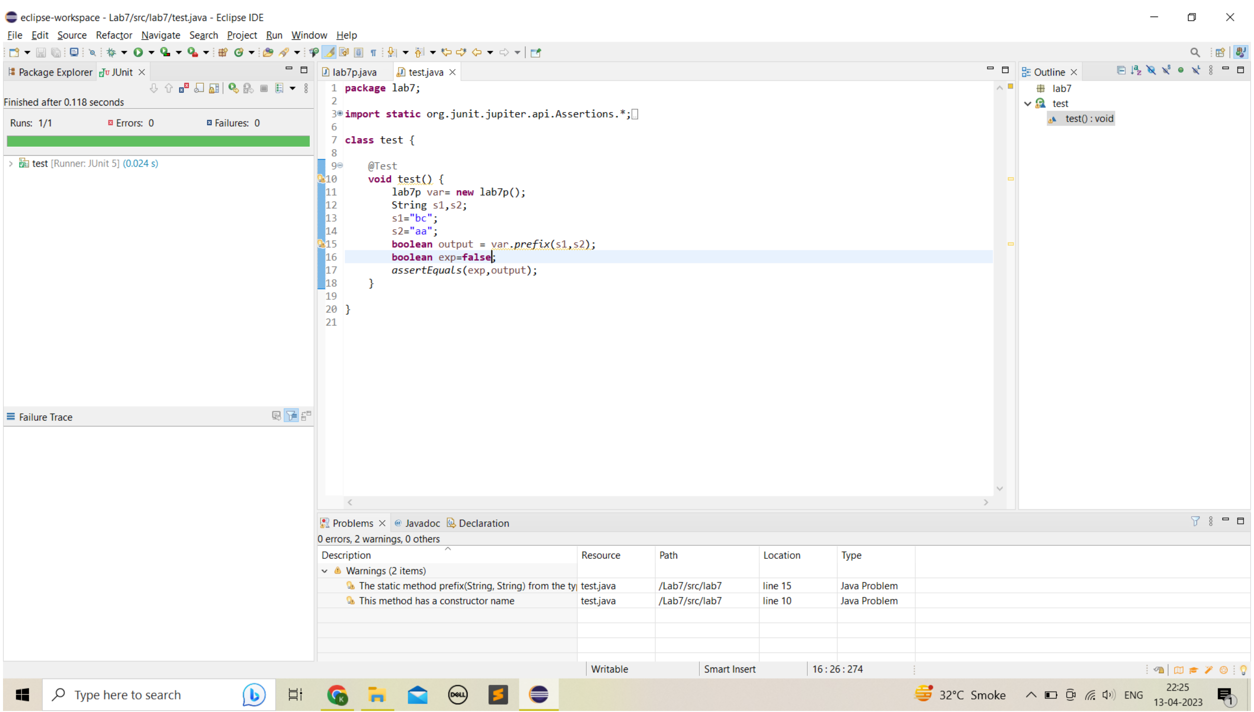
s1	s2	Expected Output
a	abc	true
<null>	abc	true
<null>	<null>	true

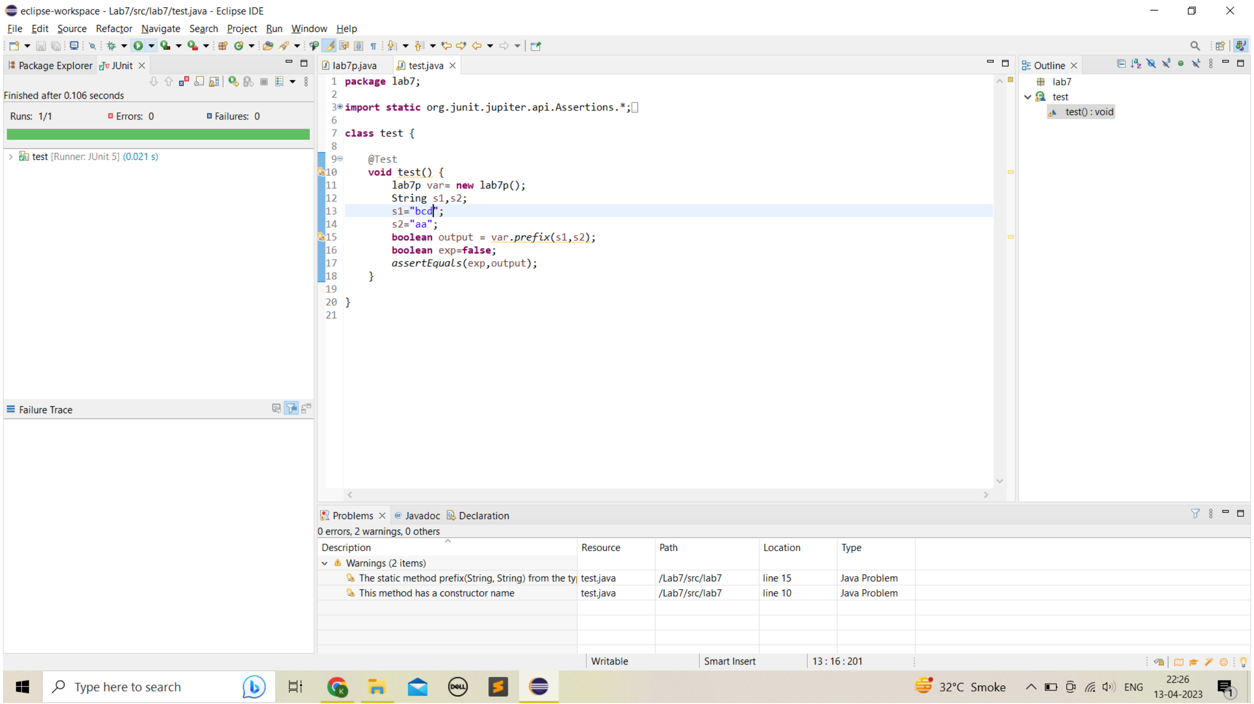
bc	aa	false
bcd	aa	false
bcd	aabcd	false

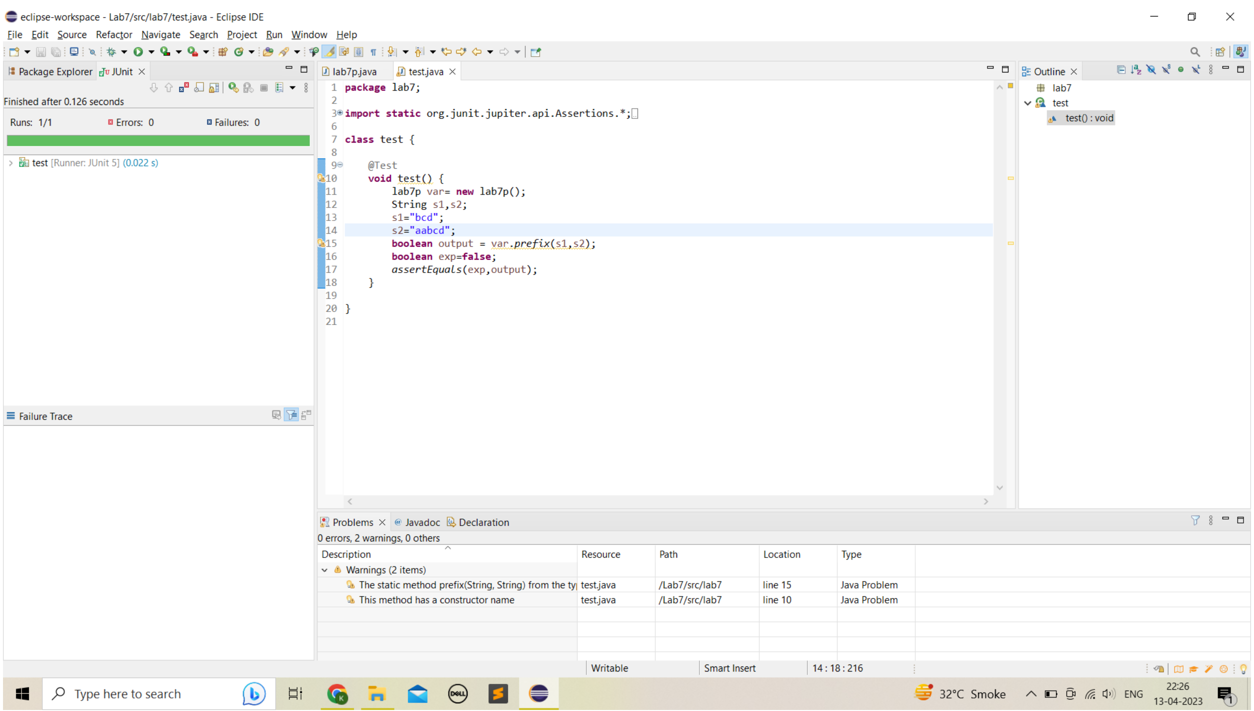












P6:

Consider again the triangle classification program (P4) with a slightly different specification: The program reads floating values from the standard input. The three values A, B, and C are interpreted as representing the lengths of the sides of a triangle. The program then prints a message to the standard output that

states whether the triangle, if it can be formed, is scalene, isosceles, equilateral, or right angled. Determine the following for the above program:

a) Identify the equivalence classes for the system

Equivalence Classes will contain:

1. **All sides are positive, real numbers.**
2. **One or more sides are negative or zero.**
3. **The sum of the lengths of any two sides is less than or equal to the length of the remaining side.**
4. **The sum of the lengths of any two sides is greater than the length of the remaining side.**

Examples

EC1 : a=c, a!=b
EC2 : a=b, b=c,c=a
EC3 : a=b, a!=c
EC4 : b=c, b!=a
EC5 : b+c<=a
EC6 : a+c<=b
EC7 : a+b<=c
EC8 : a!=b ,b!=c, c!=a
EC9 : a^2 + b^2 = c^2
EC10: b^2 + c^2 = a^2
EC11: a^2 + c^2 = b^2
EC12 : a+b>=c
EC13 : a+c>=b
EC14 : b+c>=a
EC15 : a>0 b>0 c>0
EC16 : a<=0 b=0 c>0

b) Identify test cases to cover the identified equivalence classes. Also, explicitly mention which test case would cover which equivalence class.

- Test cases

1. Right angled triangle : A = 3, B = 4, C = 5
2. Equilateral triangle : A = 1, B = 1, C = 1
3. Scalene triangle : A = 2, B = 4, C = 2
4. Isosceles triangle : A = 1, B = 2, C =3
5. Invalid Input : A = 1, B = 2, C = 9

6. Invalid Input : A = 0, B = 3, C = -1

c) For the boundary condition A + B > C case (scalene triangle), identify test cases to verify the boundary.

- Test cases

1. A = 3, B = 4, C =5
2. A = 5, B = 2, C = 7
3. A = 2, B = 1, C = 6

d) For the boundary condition A = C case (isosceles triangle), identify test cases to verify the boundary.

- Test cases

1. A = 2, B = 3, C = 2
2. A = 2, B = 3, C = 2.1
3. A = 2, B = 3, C =3

e) For the boundary condition A = B = C case (equilateral triangle), identify test cases to verify the boundary.

- Test cases

1. A = 1, B = 1, C = 1
2. A = 2, B = 1.9, C = 2.1
3. A = 1, B = 1, C = 1.2

f) For the boundary condition A2 + B2 = C2 case (right-angle triangle), identify test cases to verify the boundary.

- Test cases

1. A = 3, B = 4, C = 5
2. A = 6, B = 5, C = 10

g) For the non-triangle case, identify test cases to explore the boundary.

- Test cases

1. A = 2, B = 2, C = 4
2. A = 2, B = 4, C = 1
3. A = 2, B = 3, C = 6

h) For non-positive input, identify test points.

- Test cases

1. A = -3, B = 1, C =0
2. A = 0, B = 1, C = 3

Section-B :

The code below is part of a method in the ConvexHull class in the VMAP system. The following is a small fragment of a method in the ConvexHull class. For the purposes of this exercise you do not need to know the intended function of the method. The parameter p is a Vector of Point objects, p.size() is the size of the vector p, (p.get(i)).x is the x component of the i

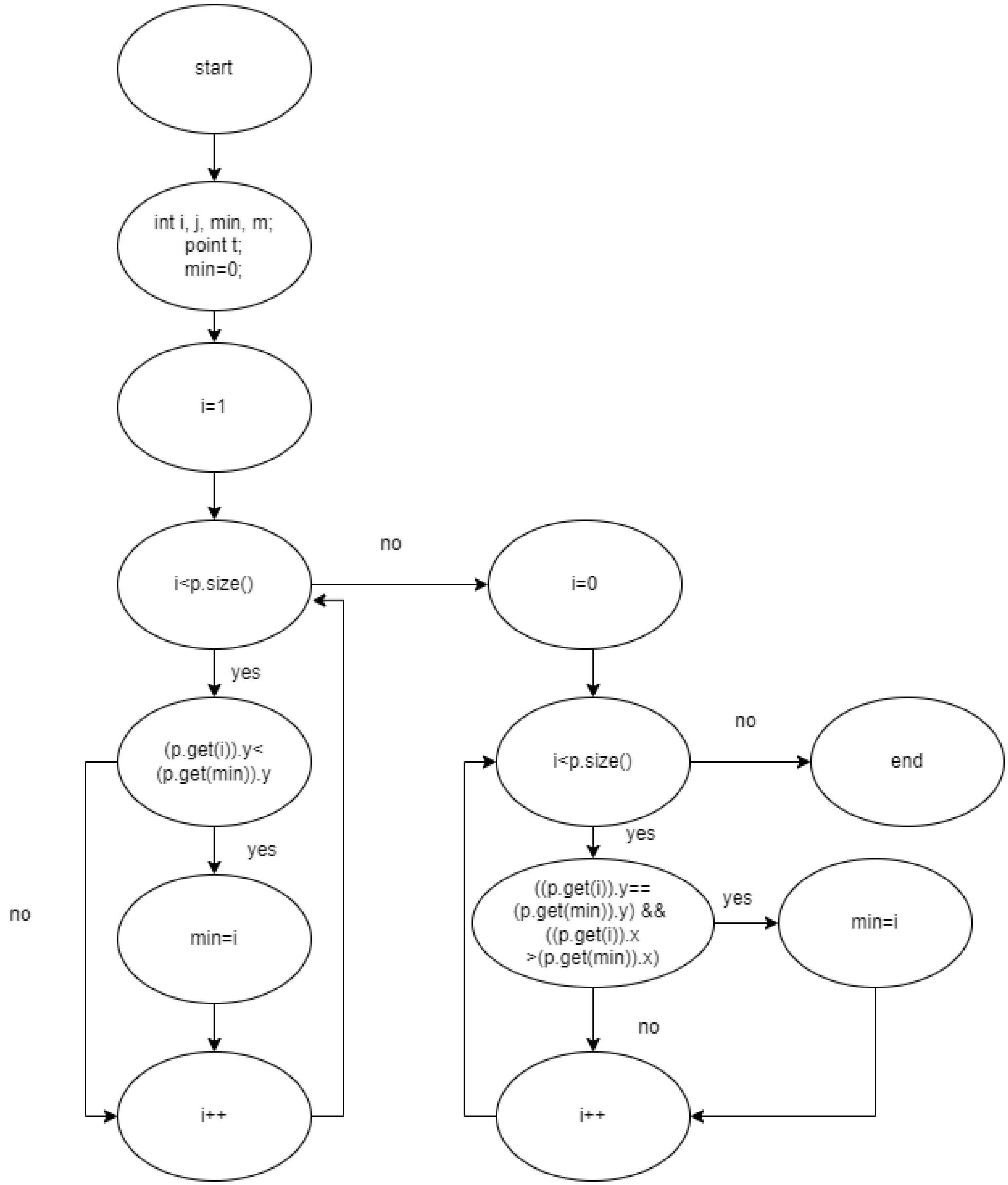
th point appearing in p, similarly for (p.get(i)).y. This exercise is concerned with structural testing of code and so the focus is on creating test sets that satisfy some particular coverage criterion.

For the given code fragment you should carry out the following activities.

```java
Vector doGraham(Vector p) {
        int i,j,min,M;

        Point t;
        min = 0;

        // search for minimum:
        for(i=1; i < p.size(); ++i) {
            if( ((Point) p.get(i)).y <
                          ((Point) p.get(min)).y )
            {
                min = i;
            }
        }

        // continue along the values with same y component
        for(i=0; i < p.size(); ++i) {
            if(( ((Point) p.get(i)).y ==
                           ((Point) p.get(min)).y ) &&
                 (((Point) p.get(i)).x >
                           ((Point) p.get(min)).x ))
            {
                 min = i;
            }
        }
```

1. Convert the Java code comprising the beginning of the doGraham method into a control flow graph (CFG).

1. **Control Flow Graph**



2. **Construct test sets for your flow graph that are adequate for the following criteria: a. Statement Coverage. b. Branch Coverage. c. Basic Condition Coverage.**

a) Statement coverage test sets:

- Test cases

1. p is an empty vector
2. p is a vector with one point
3. p is a vector with two or more points
    - 3.1 same points
    - 3.2 different points
        - 3.1.1 same y component
            - Same x,different x
        - 3.1.2 different y component
            - Same x,different x
        - 3.1.3 same x component
            - Same y,different y
        - 3.1.4 different x component
            - Same y,different y

b) Branch coverage test sets:

- Test cases

    1. p is an empty vector

    2. p is a vector with one point

    3. p is a vector with two or more points
        - 3.1 same points
        - 3.2 different points
            - 3.1.1 same y component
                - Same x,different x
            - 3.1.2 different y component
                - Same x,different x
            - 3.1.3 same x component
                - Same y,different y
            - 3.1.4 different x component
                - Same y,different y

c) Basic condition coverage test sets:

- Test cases

1. p is an empty vector
2. p is a vector with one point
3. p is a vector with two or more points

3.1 same points
3.2 different points
3.1.1 same y component
Same x,different x
3.1.2 different y component
Same x,different x
3.1.3 same x component
Same y,different y
3.1.4 different x component
Same y,different y

- Test cases examples:

1. p=[]

2. p=[(1,1)]

3. p=[(1,1)(1,1)]

4. p=[(1,2)(1,2)(1,1)(3,4)]

5. p=[(1,3)(2,4)]

These 5 test cases cover all - statement coverage, branch coverage and basic condition coverage.